CSCI 465: Web Programming Fundamentals
Assignment p1: rfind (Ruby on Rails find utility)

Write a ruby script that takes a single regular expression as a command line argument and searches the current directory and all its subdirectories for (1) files with names that match the given regular expression and (2) files that have content that matches the given regular expression.

This utility is very helpful when learning to program in Rails. Part of the challenge is figuring out what goes in which file.

For files that have content matching the regular expression, print the filename and then every line (with line numbers) that match the regular expression.

Use the format in the following examples so I can use automated testing on your programs. The output from the second example was truncated.

```
$ rfind "exit_status.*="
Files with names that matches <exit_status.*=>
************************************************
Files with content that matches <exit_status.*=>
./db/schema.rb
  62:    t.boolean  "ignore_exit_status",   :default => true
  63:    t.integer  "exit_status",          :default => 0
------------------------------------------------
./db/migrate/20110927191314_add_exit_status_to_grading_cases.rb
  3:    add_column :grading_cases, :ignore_exit_status, :boolean, :default => true
  4:    add_column :grading_cases, :exit_status, :integer, :default => 0
------------------------------------------------
./app/models/grading_case.rb
  125:          exit_status_file = File.open(base + '.exit', "r")
  126:          new_grading_case.exit_status = exit_status_file.readline.to_i
  127:          new_grading_case.ignore_exit_status = false
  131:          new_grading_case.ignore_exit_status = true
------------------------------------------------
./app/models/submission.rb
  409:      process_exit_status = $?.exitstatus
  417:      if process_exit_status == 101
  422:      elsif process_exit_status == 100
  435:      elsif process_exit_status == 139
  450:      if process_exit_status != 139
$
$
$
$ rfind grading_result
Files with names that matches <grading_result>
  ./db/migrate/20120720205256_add_stdout_size_to_grading_results.rb
  ./db/migrate/20110721223154_create_grading_results.rb
  ./db/migrate/20110722162642_add_execution_status_to_grading_results.rb
  ./db/migrate/20110725220144_add_correct_exit_status_to_grading_results.rb
  ./app/helpers/grading_results_helper.rb
  ./app/models/grading_result.rb
  ./app/controllers/grading_results_controller.rb
  ./app/views/grading_results/new.html.erb
  ./app/views/grading_results/edit.html.erb
  ./app/views/grading_results/_form.html.erb
  ./app/views/grading_results/show.html.erb
  ./app/views/grading_results/index.html.erb
  ./test/functional/grading_results_controller_test.rb
  ./test/fixtures/grading_results.yml
  ./test/unit/helpers/grading_results_helper_test.rb
  ./test/unit/grading_result_test.rb
************************************************
Files with content that matches <grading_result>
./db/schema.rb
  70:  create_table "grading_results", :force => true do |t|
------------------------------------------------
./db/migrate/20120720205256_add_stdout_size_to_grading_results.rb
  3:    add_column :grading_results, :stdout_size, :integer, :default => 0
  4:    add_column :grading_results, :stderr_size, :integer, :default => 0
  10:    remove_column :grading_results, :stdout_size
  11:    remove_column :grading_results, :stderr_size
------------------------------------------------
./db/migrate/20110721223154_create_grading_results.rb
```

```
  3:    create_table :grading_results do |t|
  16:    drop_table :grading_results
--------------------------------------------------
./db/migrate/20110722162642_add_execution_status_to_grading_results.rb
  3:    add_column :grading_results, :execution_status, :string
  7:    remove_column :grading_results, :execution_status
```

Formatting note:

The filenames (in the first section) are indented 2 spaces (may be hard to tell with .html).

The matching lines from the file are indented with 2 spaces. All spaces after the "line_number:" are from the file itself (that is, don't add spaces).

The dividers (------ and ****) each have 50 characters.

Requirements

Put the following line as the first line (MUST BE FIRST LINE, DO NOT PUT COMMENT BEFORE IT) of rfind.rb. Don't indent it.

#!/usr/bin/env ruby

The regular expression should be case insensitive. (Usually you have to put the regular expression in quotes so the bash shell does not interpret it before passing it to your program.)

Only search files the the following extensions: .rb .erb .js .css .html .yml .txt

Print the line numbers when printing a line that matches the regular expression.

You may use standard Linux utilities (such as **find** and **grep**).

I have posted two tests and some sample directories with sample files. The first test is "$ rfind chico" the second test is "$ rfind 1.*nia$"

How to test:

download the p1.tar file
$ tar -xvf p1.tar
$ cd p1
$ rfind chico > chico.myout
compare your chico.myout to the posted chico.out

$ rfind "1.*nia$" > regex.myout <-- that is a ONE, not an L
compare your regex.myout to the posted regex.out

Turnin the single file rfind.rb to turnin.

You may turn the assignment in up to 24 hours late for a 10% penalty.

Hints

If you use grep to search the contents of files, you can use the -i option for case insensitivity and -n for line numbers.

Using the Linux find utility will save some time. However, you are not required to use it. Ruby has libraries for reading directories.

Use backtics to get the results of a command. For example:

files = `find .`

The variable file is a string containing the results of the command `find .`

Now you can use the String split function to turn this single string into an array and then the Array map function to find all the elements of the array that match the regular expression.

grep will handle the regular expressions for you. The regular expressions used by grep are different that the sort-of-regular expressions used

by bash.

WARNING: If you find yourself using the -r (recursive) option for grep, you are heading down a difficult road. It is much easier to use find to build a list of file names and then pass these names to grep one at a time.

ruby-doc.org provides a great reference.